Piyavskii's algorithm

for deterministic or stochastic Lipschitz bandit optimization

Clément Bouttier and Sébastien Gerchinovitz

Thuesday 24th Mai, Erasmus Universty Rotterdam

Few words about my PhD thesis

Global optimization under uncertainty: stochastic algorithms and bandits convergence bounds with application to aircraft performance

Supervision

University: Pr. Sébastien Gadat, Asst. Sébastien Gerchinovitz, Asst. Florence Nicol

Airbus, Aircraft Performance Department: Olivier Babando, Serge Laporte

Digital Transformation Office

Airline Sciences Group

Agenda

1. Introduction

2. Piyavskii Algorithm

3. Stochastic Piyavskii Algorithm

Introduction

A famous problem

Problem

$$\begin{aligned} \text{Find} \quad & x_n^\star \in \mathcal{X} = [0,1]^d \\ \text{such that} \quad & f(x_n^\star) \geqslant \max_{x \in \mathcal{X}} f(x) - \varepsilon \end{aligned}$$

given a minimal set of sequential observations:

$$f(x_1), f(x_2), \ldots, f(x_n)$$

A famous problem

Problem

$$\text{Find} \quad x_n^\star \in \mathcal{X} = [0,1]^d$$
$$\text{such that} \quad f(x_n^\star) \geqslant \max_{x \in \mathcal{X}} f(x) - \varepsilon$$

given a minimal set of sequential observations:

$$f(x_1), f(x_2), \ldots, f(x_n)$$

Hereafter we also consider the Sub-Gaussian context

$$f(x_k) + \xi_k \text{ where } \xi_k \text{ is sub-Gaussian instead of } f(x_k)$$

$\mathcal{X} = [0,1]^d$, a regularity assumption is needed

"Local" Lipschitz, regularity assumption
There exists $L_0 > 0$ such that

$$\forall x \in [0,1]^d,\ f(x) \geqslant f(x^\star) - L_0 \|x^\star - x\| .$$

"Optimistic" Lipschitz Bandit optimization process

At every step perform:

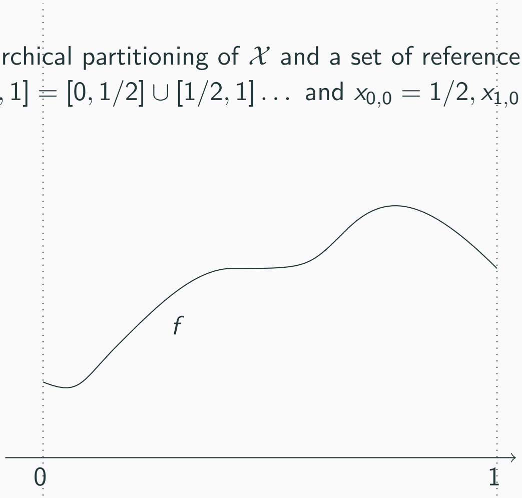
Upper bound generation
Build $\widehat{f}$ an upper-bound of f using both the regularity assumption on f and about the noise

Maximization
Choose evaluation points according to an optimistic principle *i.e.*, choose the maximizer of $\widehat{f}$

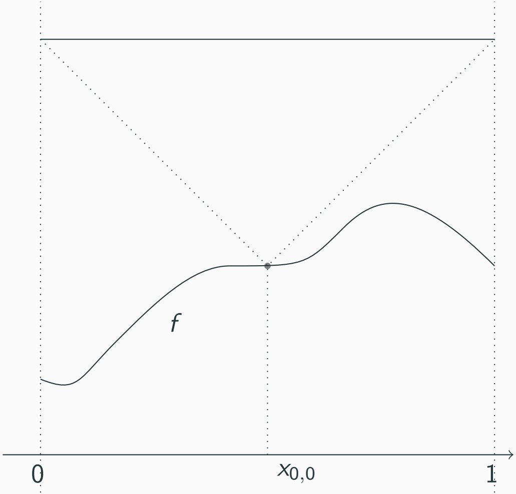
Deterministic Optimistic Optimization (DOO) by Munos [2011]

Hierarchical partitioning of $\mathcal{X}$ and a set of reference points

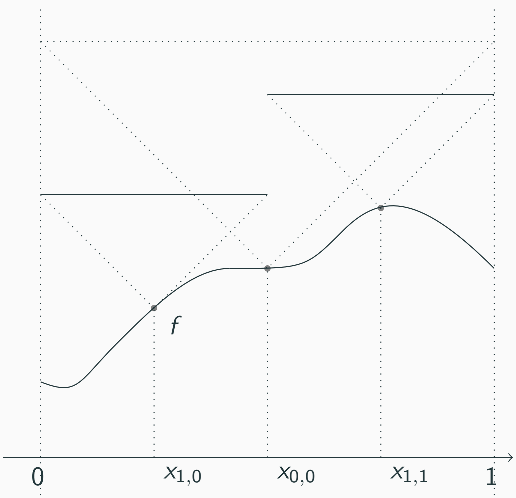
ex : binary tree : $([0,1] = [0,1/2] \cup [1/2,1] \dots$ and $x_{0,0} = 1/2, x_{1,0} = 1/4, x_{1,1} = 3/4 \dots)$



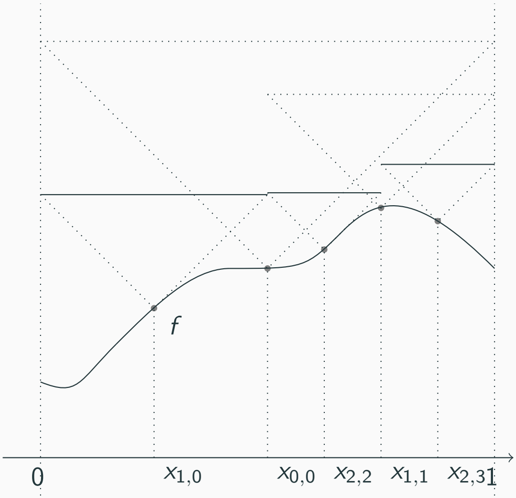
Deterministic Optimistic Optimization (DOO) by Munos [2011]



Deterministic Optimistic Optimization (DOO) by Munos [2011]



Deterministic Optimistic Optimization (DOO) by Munos [2011]



Convergence ?

$$x^\star \in \arg\max_{x \in \mathcal{X}} f(x)$$

$x_n^\star$ recommendation of the algorithm

Definition (Simple regret = optimization error)

$$r_n = f(x^\star) - f(x_n^\star)$$

Objectiv: upper bound on r_n

Measuring the difficulty of the problem

Let $\varepsilon > 0$,

$$\mathcal{X}_\varepsilon = \left\{x \in [0,1]^d, f(x) \geqslant f(x^\star) - \varepsilon\right\}$$

$$\text{Packing Number} \quad \mathcal{N}_L(A, \varepsilon) := \sup\left\{k \in \mathbb{N}^* : \exists x_1, \ldots, x_k \in A, \min_{i \neq j} \|x_i - x_j\| > \varepsilon / L\right\}$$

$$\varepsilon_0 := L_0 \sup_{x,y \in [0,1]^d} \|x - y\| .$$

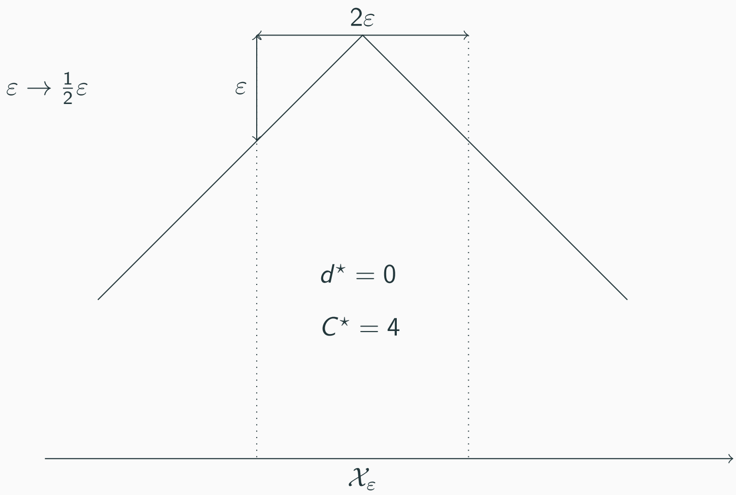
Near-Optimality Dimension

There exists $d^\star \in [0, d]$ and $C^\star \geqslant 0$ such that

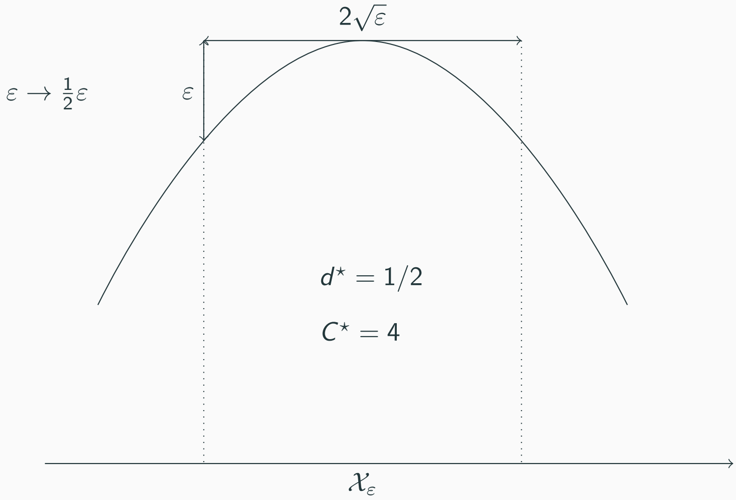
$$\forall \varepsilon \in (0, \varepsilon_0], \quad \mathcal{N}_{L_0}\left(\mathcal{X}_\varepsilon, \frac{1}{2}\varepsilon\right) \leqslant C^\star \left(\frac{\varepsilon_0}{\varepsilon}\right)^{d^\star}$$

The smallest $d^\star$ is called "Near-Optimality Dimension"

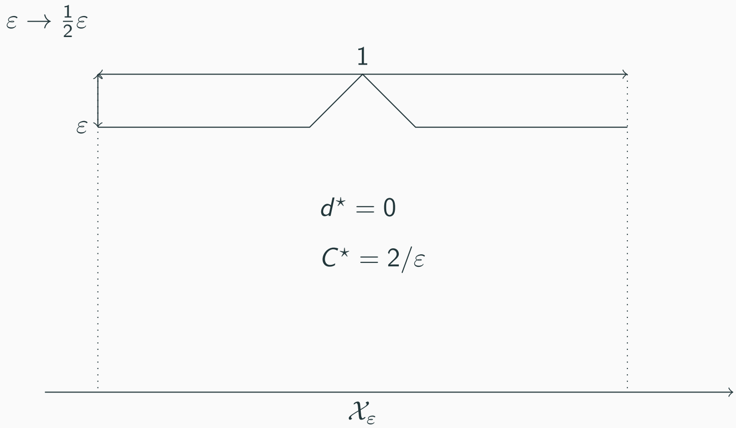
Getting a feeling for Near-Optimality dimension



Getting a feeling for Near-Optimality dimension



Getting a feeling for Near-Optimality dimension



Regret bound for DOO

Assumption

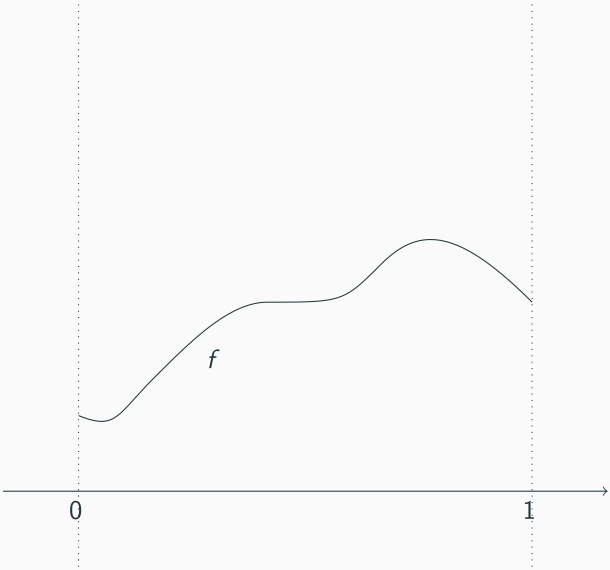
- *L and $\|.\|$ known*
- *k-adic partitioning adapted to L and $\|.\|$*

Theorem (Munos [2011])

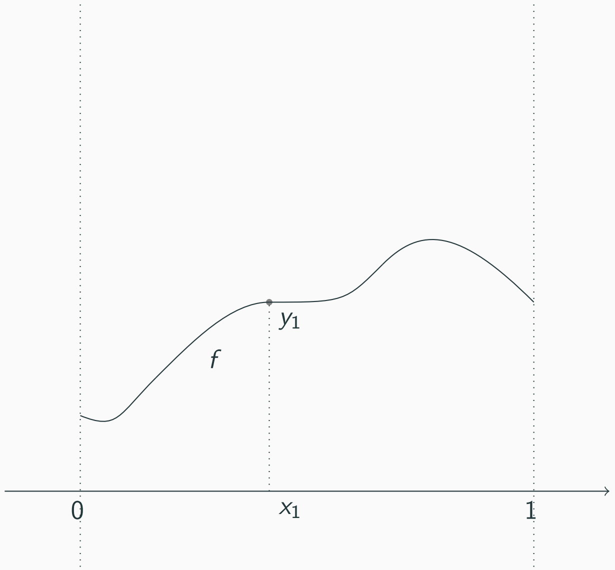
$$r_n(DOO) \leqslant \begin{cases} \mathcal{O}\left(e^{\frac{-n\ln(k)}{C^\star}}\right) & \textit{if } d^\star = 0 \\ \mathcal{O}\left(C^{\star\frac{1}{d^\star}} n^{-\frac{1}{d^\star}}\right) & \textit{if } d^\star > 0 \end{cases}$$

Piyavskii Algorithm

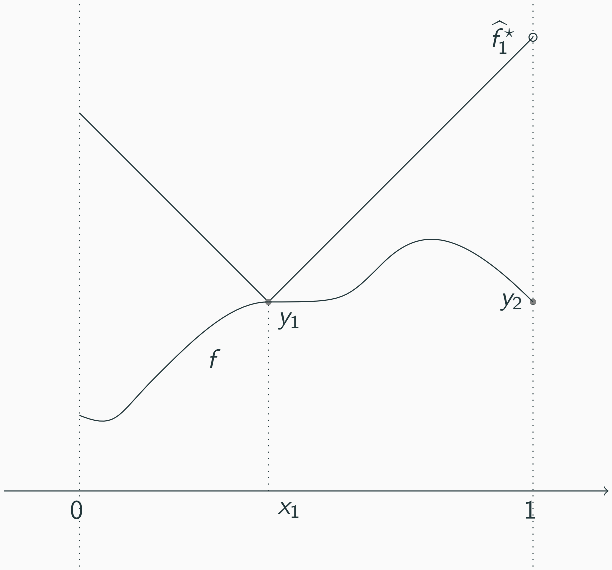
Why using a piece-wise constant partitioning?



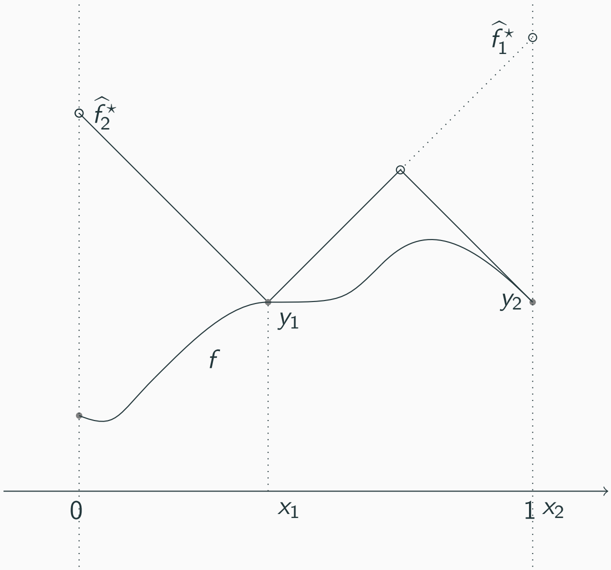
Why using a piece-wise constant partitioning?



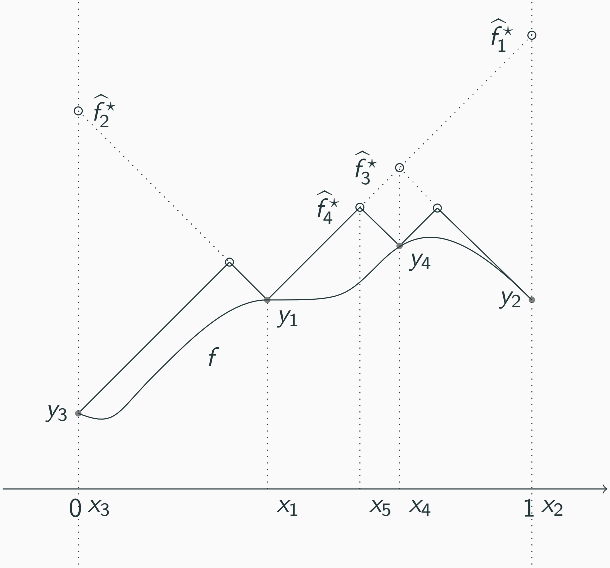
Why using a piece-wise constant partitioning?



Why using a piece-wise constant partitioning?



Why using a piece-wise constant partitioning?



Piyavskii's Algorithm [Piyavskii, 1972]

Algorithm 1 PY(known budget n) ([Piyavskii, 1972])

Inputs: Lipschitz Constant L_1, evaluation number n, initial solution $x_1 \in [0,1]^d$

for k from 1 to n **do**

Observe $y_k = f(x_k)$

Update $\widehat{f}_k(x) = \min_{0 \leqslant i \leqslant k} \{ y_i + L_1 \|x_i - x\| \}$ for all $x \in [0,1]^d$

Determine $x_{k+1} = \arg\max_{x \in [0,1]^d} \widehat{f}_k(x)$

end for

return $x_n^\star := \arg\max_{x \in \{x_1, \ldots, x_n\}} f(x)$

Regret bound [B. et Gerchinovitz, 2017]

Theorem ([B. et Gerchinovitz, 2017])

$$r_n \leqslant \begin{cases} C_1 \ 2^{-n \ C_2} \ \textit{if } d^\star = 0 \\ C_3 \ n^{-1/d^\star} \ \textit{if } 0 < d^\star \leqslant d \end{cases}$$

Regret bound [B. et Gerchinovitz, 2017]

Theorem ([B. et Gerchinovitz, 2017])

$$r_n \leqslant \begin{cases} \varepsilon_0 2^{1-\frac{n-1}{C^\star (1+4L_1/L_0)^d}} \text{ si } d^\star = 0 \\ \varepsilon_0 \left(1-2^{-d^\star}\right)^{-\frac{1}{d^\star}} C^{\star\frac{1}{d^\star}} \left(1+\frac{4L_1}{L_0}\right)^{\frac{d}{d^\star}} (n-1)^{-1/d^\star} \text{ si } 0 < d^\star \leqslant d \end{cases}$$

Main elements of the proof

(1) Let $\Delta > 0$. If $x_i \in \mathcal{X}_\Delta^c$, then $\forall j > i$, then $\|x_j - x_i\| > \frac{\Delta}{L_1}$

Regret bound [B. et Gerchinovitz, 2017]

Theorem ([B. et Gerchinovitz, 2017])

$$r_n \leqslant \begin{cases} \varepsilon_0 2^{1-\frac{n-1}{C^\star(1+4L_1/L_0)^d}} & si\ d^\star = 0 \\ \varepsilon_0 \left(1-2^{-d^\star}\right)^{-\frac{1}{d^\star}} C^{\star\frac{1}{d^\star}} \left(1+\frac{4L_1}{L_0}\right)^{\frac{d}{d^\star}} (n-1)^{-1/d^\star} & si\ 0 < d^\star \leqslant d \end{cases}$$

Main elements of the proof

(1) Let $\Delta > 0$. If $x_i \in \mathcal{X}^c_\Delta$, then $\forall j > i$, then $\|x_j - x_i\| > \frac{\Delta}{L_1}$

(2) Peeling technique :

$$\mathrm{card}(\{k \in \{1,\ldots,n\} : x_k \in \mathcal{X}^c_\varepsilon\}) \leqslant \sum_{s=1}^{m_\varepsilon} \mathrm{card}\left(\left\{k \in \{1,\ldots,n\} :\ x_k \in \mathcal{X}_{(\varepsilon_0 2^{-s},\varepsilon_0 2^{-s+1}]}\right\}\right)$$

What about retrieving an ε optimal solution?

We can determine the minimal number of sampling $\bar{n}_{PY}$ to perform to ensure $x_n^\star \in \mathcal{X}_\varepsilon$.

If $0 < d^\star \leqslant d$,

$$\bar{n}_{PY}(\varepsilon, d^\star, C^\star) \simeq C^\star \varepsilon^{-d^\star}$$

But $C^\star$ and $d^\star$ are unkwon...

Piyavskii's Algorithm with automatic stopping

Algorithm 2 PY(with given final precision ε)

Inputs: precision $\varepsilon > 0$, Lipschitz constant L_1, initial guess $x_1 \in [0,1]^d$

Execute PY(0)

$k = 0$

while $\widehat{f}_k^\star - f_k^\star > \varepsilon$ **do**

  Make an additional PY iteration

  $k = k + 1$

end while

return $x_k^\star := x_{i_k^\star}$

Number of evaluations

Corollary ([B. and Gerchinovitz, 2017])

$$n_{\varepsilon PY}(\varepsilon, d^\star, C^\star) \leqslant \begin{cases} \frac{3}{2}\bar{n}_{PY}(\varepsilon/2, d^\star, C^\star) \ si\ d^\star = 0 \\ 2\bar{n}_{PY}(\varepsilon/2, d^\star, C^\star) \ si\ d^\star > 0 \end{cases}$$

Hansen et al. [1992]

Problem number	$\varepsilon = 10^{-7}\gamma$, i.e., $n_{pass} = 10^7$								
	n_B	$\frac{n_{pass}}{n_B}$	$\frac{n_{EV}}{n_B}$	$\frac{n_{GA}}{n_B}$	$\frac{n_{SZ}}{n_B}$	$\frac{n_{PY}}{n_B}$	$\frac{n_{TM}}{n_B}$	$\frac{n_{SC}}{n_B}$	$\frac{n_{new}}{n_B}$
1	3 415	2928	1893	3.980	2.988	1.424	1.345	1.316	1.014
2	2 724	3671	811	3.977	2.986	1.445	1.341	1.337	1.017
3	3 148	3177	79	3.968	2.993	1.448	1.345	1.330	1.030
4	8 533	1172	568	3.982	2.988	1.495	1.365	1.331	1.007
5	2 460	4065	402	3.976	2.986	1.488	1.364	1.347	1.015
6	1 887	5299	1864	3.976	2.974	1.482	1.360	1.337	1.020
7	3 223	3103	889	3.977	2.989	1.488	1.345	1.318	1.016
8	2 979	3357	115	3.973	2.988	1.462	1.365	1.332	1.026
9	2 650	3774	725	3.980	2.990	1.376	1.346	1.322	1.011
10	3 650	2740	945	3.981	2.989	1.480	1.353	1.345	1.007
11	7 092	1410	89	3.982	2.988	1.489	1.364	1.339	1.009
12	6 789	1473	458	3.982	2.989	1.421	1.354	1.328	1.010
13	10 817	924	327	3.983	2.989	1.377	1.346	1.322	1.005
14	2 255	4435	250	3.977	2.987	1.483	1.350	1.345	1.022
15	14 549	687	121	3.981	2.988	1.365	1.346	1.321	1.006
16	9 201	1087	832	3.981	2.988	1.467	1.356	1.345	1.006
17	12 013	832	105	3.980	2.988	1.364	1.346	1.322	1.007
18	5 736	1743	582	3.981	2.989	1.490	1.361	1.339	1.006
19	2 678	3734	1416	3.980	2.990	1.416	1.361	1.328	1.010
20	5 084	1967	733	3.929	2.972	1.459	1.345	1.344	1.031
mean value	5544	2579	660	3.975	2.987	1.446	1.353	1.332	1.014
deviation	3709	1369	554	0.013	0.005	0.046	0.008	0.010	0.008

Figure 1: Comparison of numerical performances for 1D Lipschitz optimization

Numerical performance of PY vs. DOO

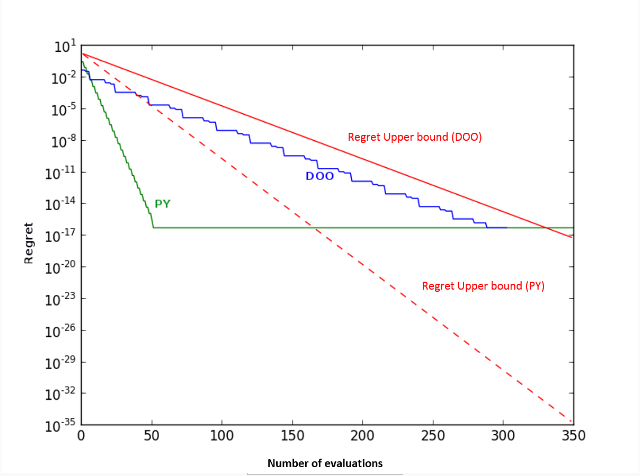


Figure 2: $d^\star = 0$

Numerical performance of PY vs. DOO

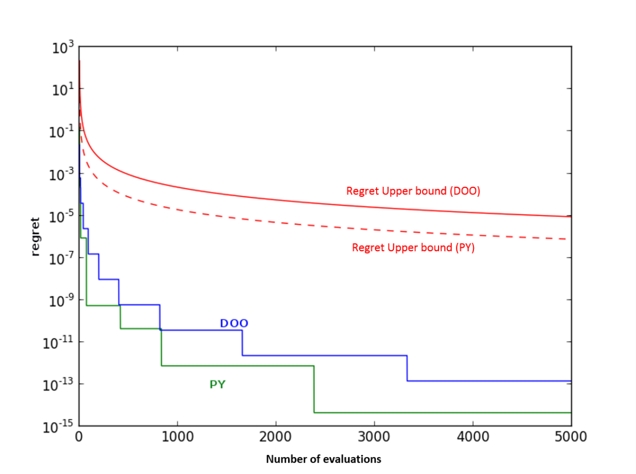


Figure 2: $d^\star = 1/2$

Stochastic Piyavskii Algorithm

Regret bound for a Stochastic extension of DOO: StoOO

Assumption

- *L and $\|.\|$ known*
- *Partitioning adapted to L and $\|.\|$*
- *ξ sub-Gaussian*

Theorem ([Valko et al., 2013], [Munos, 2014])

$$r_n(StoOO) \leqslant \left\{ \mathcal{O}\left(C^{\star\,\frac{1}{d^\star}} n^{-\frac{1}{d^\star+2}} \right) \text{ if } d^\star \geqslant 0 \right.$$

Stochastic Extension of Piyavskii's Algorithm (SPY)

Algorithm 3 SPY(known budget n)

Inputs: Lipschitz constant L_1, evaluation number n, mini-batch's size $n_B \leqslant n$, risk level $\delta \in (0,1]$, initial guess $x_1 \in [0,1]^d$

Set $N = \lfloor n/n_B \rfloor$ and $\delta_H = \sqrt{\frac{2\sigma^2 \ln(2n\delta^{-1})}{n_B}}$

for k from 1 to N **do**

Sample n_B times in x_k, and collect $\left(Y_i^k\right)_{1 \leqslant i \leqslant n_B}$, where $Y_i^k = f(x_k) + \xi_i^k$

Set $y_k = \frac{1}{n_B} \sum_{i=1}^{n_B} Y_i^k$

Update $\widehat{f}_k(x) = \min_{j \in \{1,\ldots,k\}} \{y_j + L_1 \|x_j - x\| + \delta_H\}$

Determine $x_{k+1} = \arg\max_{x \in [0,1]^d} \widehat{f}_k(x)$

end for

Determine $i_N^\star = \arg\max_{1 \leqslant i \leqslant N} y_i$

return $x_N^\star := x_{i_N^\star}$

Controlling the noise

We assume iid sub-Gaussian noise, ie:

$$\max\left(\mathbb{P}\left(\xi_i > x\right), \mathbb{P}\left(\xi_i < -x\right)\right) \leqslant e^{-x^2/(2\sigma^2)}.$$

and thus:

$$\forall x_k \in [0,1]^d, \forall s \geqslant 0, \forall n_B \in \mathbb{N}^* \quad \mathbb{P}\left(\left|\frac{1}{n_B}\sum_i^{n_B} Y_i^k - f(x_k)\right| \geqslant x\right) \leqslant 2e^{-n_B s^2/(2\sigma^2)}$$

Regret bound for SPY

Known budget

Theorem ([B. and Gerchinovitz, 2017])
If $n_B \simeq \frac{\ln(2n/\delta)}{\varepsilon^2}$ *and*

$$\frac{n}{\varepsilon_0^2 + 128\sigma^2 \ln\left(2n\delta^{-1}\right)} > \begin{cases} \mathcal{O}\left(\varepsilon^{-2}\right) & \text{if } d^\star = 0 \\ \mathcal{O}\left(\varepsilon^{-d^\star-2}\right) & \text{if } 0 < d^\star \leqslant d \end{cases}$$

then

$$\mathbb{P}\left(f(x_N^\star) \geqslant f(x^\star) - \varepsilon\right) \geqslant 1 - \delta$$

Conclusion

Conclusion

- Mind the gap between traditional optimization community and the bandit community by providing modern assessment of (well known) algorithms
- Introduce simple element of proof adapted to higher dimensions and stochastic extensions

Perspectives

- Update $\widehat{f}_k(x) = \min_{0 \leqslant i \leqslant k} \{y_i + L_1 \|x_i - x\|\}$ for all $x \in [0,1]^d$
- Numerical assessment on real problems

Thank you for your attention.

Annexes

References

Pierre Hansen, Brigitte Jaumard, and Shi-Hui Lu. Global optimization of univariate lipschitz functions: Ii. new algorithms and computational comparison. Mathematical programming, 55 (1):273–292, 1992.

Rémi Munos. Optimistic optimization of a deterministic function without the knowledge of its smoothness. In NIPS, pages 783–791, 2011.

Rémi Munos. From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. 2014.

S.A. Piyavskii. An algorithm for finding the absolute extremum of a function. Comput. Math. Math. Phys., 12(4):57–67, 1972.

Michal Valko, Alexandra Carpentier, and Rémi Munos. Stochastic simultaneous optimistic optimization. In ICML (2), pages 19–27, 2013.

DOO

Algorithm 4 DOO: (Munos (2011))

Init: $\mathcal{T}_1 = [0,1]^d$ and its representant $x_{0,0}$

for t from 1 to n **do**

  Choose a Leaf (h,j) from the tree $\mathcal{T}_t$ maximising $b_{h,j} := f(x_{h,j}) + \delta(h)$

  Developp that node: add to $\mathcal{T}_t$ the k children of (h,j)

end for

return $x(n) = \arg\max_{(h,i)\in\mathcal{T}_n} f(x_{h,i})$

Recent Optimistic algorithms:

DOO, SOO, StoOO, StoSOO, HOO, POO, ...

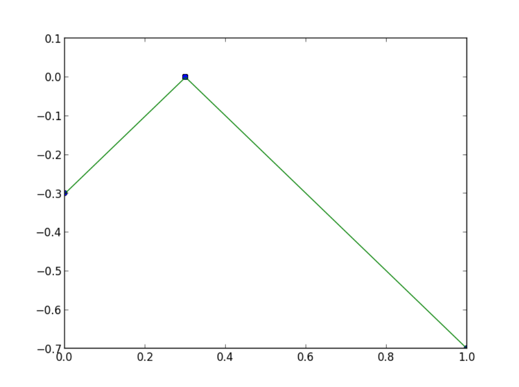
Hansen et al. [1992]

Test problems

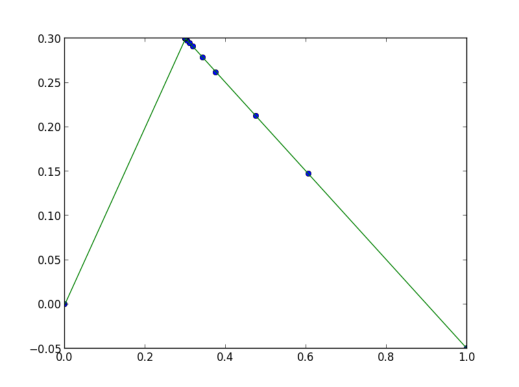
Problem number	Lipschitz function $f(x)$	Interval $[a, b]$	Lipschitz constant L	Optimum value f^*	Optimum point(s) x^*	Source from
1	$-\frac{1}{6}x^6+\frac{52}{25}x^5-\frac{39}{80}x^4-\frac{71}{10}x^3+\frac{79}{20}x^2+x-\frac{1}{10}$	$[-1.5, 11]$	13 870	29 763.233	10	[14, 24, 29]
2	$-\sin x-\sin\frac{10}{3}x$	$[2.7, 7.5]$	4.29	1.899599	5.145735	[28]
3	$\sum_{k=1}^{5} k\sin[(k+1)x+k]$	$[-10, 10]$	67	12.03125	−6.7745761 −0.49139 5.791785	[3, 14, 26, 27]
4	$(16x^2-24x+5)\,e^{-x}$	$[1.9, 3.9]$	3	3.85045	2.868	[6, 9, 14]
5	$(-3x+1.4)\sin 18x$	$[0, 1.2]$	36	1.48907	0.96609	[3, 4, 14, 26]
6	$(x+\sin x)\,e^{-x^2}$	$[-10, 10]$	2.5	0.824239	0.67956	[5]
7	$-\sin x-\sin\frac{10}{3}x$ $-\ln x+0.84x-3$	$[2.7, 7.5]$	6	1.6013	5.19978	[28]
8	$\sum_{k=1}^{5} k\cos[(k+1)x+k]$	$[-10, 10]$	67	14.508	−7.0835 −0.8003 5.48286	[18, 19, 30]
9	$-\sin x-\sin\frac{2}{3}x$	$[3.1, 20.4]$	1.7	1.90596	17.039	[28]
10	$x\sin x$	$[0, 10]$	11	7.91673	7.9787	[11]
11	$-2\cos x-\cos 2x$	$[-1.57, 6.28]$	3	1.5	2.094 4.189	[20]
12	$-\sin^3 x-\cos^3 x$	$[0, 6.28]$	2.2	1	3.142 4.712	[11]
13	$x^{2/3}-(x^2-1)^{1/3}$	$[0.001, 0.99]$	8.5	1.5874	0.7071	[11]
14	$e^{-x}\sin 2\pi x$	$[0, 4]$	6.5	0.788685	0.224885	[11]
15	$(-x^2+5x-6)/(x^2+1)$	$[-5, 5]$	6.5	0.03553	2.4142	[11]
16	$-2(x-3)^2-e^{-x^2/2}$	$[-3, 3]$	85	−7.515924	1.5907	[21]
17	$-x^6+15x^4-27x^2-250$	$[-4, 4]$	2520	−7	−3 3	[18, 19]
18	$\begin{cases}-(x-2)^2 & \text{if } x\leq 3\\ -2\ln(x-2)-1 & \text{otherwise}\end{cases}$	$[0, 6]$	4	0	2	[28]
19	$x-\sin 3x+1$	$[0, 6.5]$	4	7.81567	5.87287	[18, 19]
20	$(x-\sin x)\,e^{-x^2}$	$[-10, 10]$	1.3	0.0634905	1.195137	[5]

Figure 3: Test functions Hansen et al. [1992]

A simple example



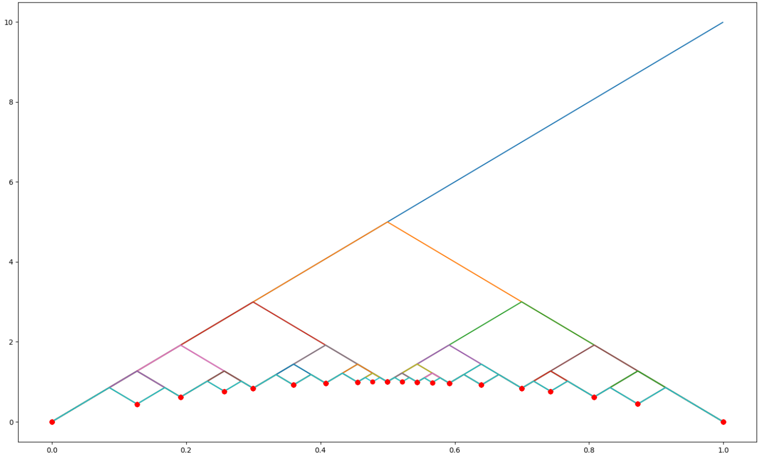
(a) $-|x - 0,3|$



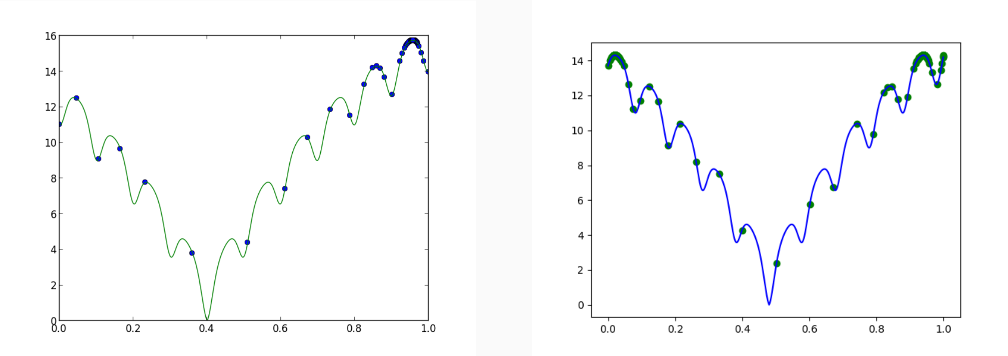
(b) $\min(x, 0.3) - 0.5 \max(x, 0.3)$

Figure 4: $(L_1 = L_0 = 1)$

Piiyavsky Algorithm vs. simple quadratic reward



$d^\star > 0$

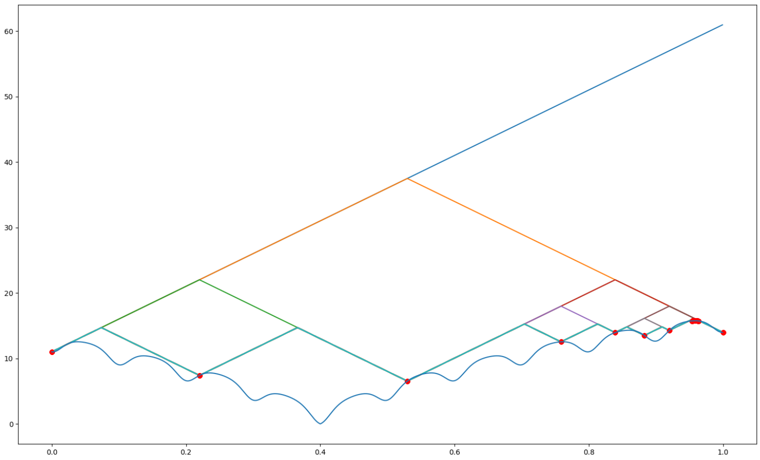


(a) Shift de 0.1

(b) Shift de 0.001

Figure 5: Ackley test function, using an upper-bound on the Lipschitz constant $L_1 = 150 > L_0$

$d^\star > 0$



$d^\star > 0$

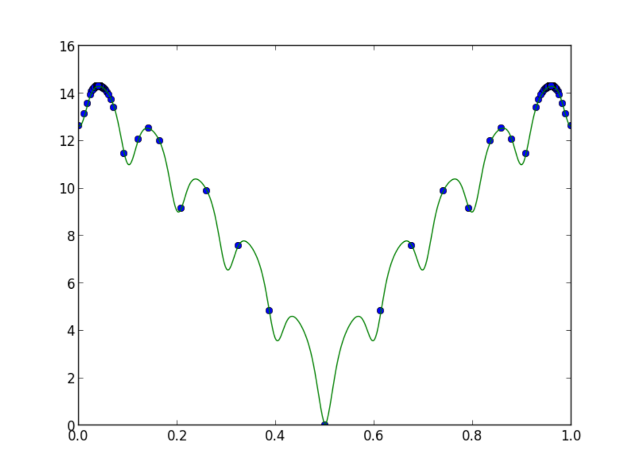
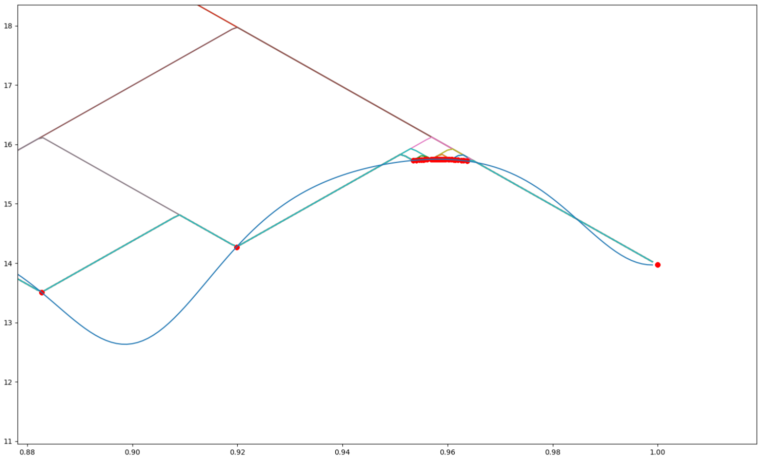


Figure 6: Ackley test function, using an upper-bound on the Lipschitz constant $L_1 = 150 > L_0$

$d^\star > 0$



$d^\star > 0$

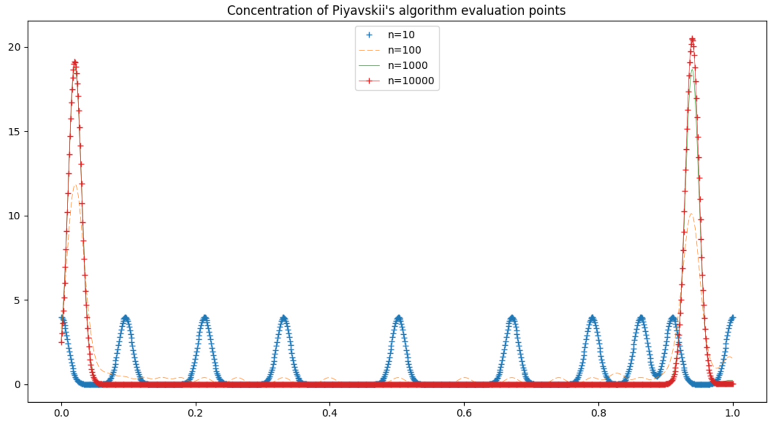


Figure 7: Kernel estimate of the density of PY evaluation points (Ackley test function with 0.001 a Shift)

$d^\star > 0$ vs DOO

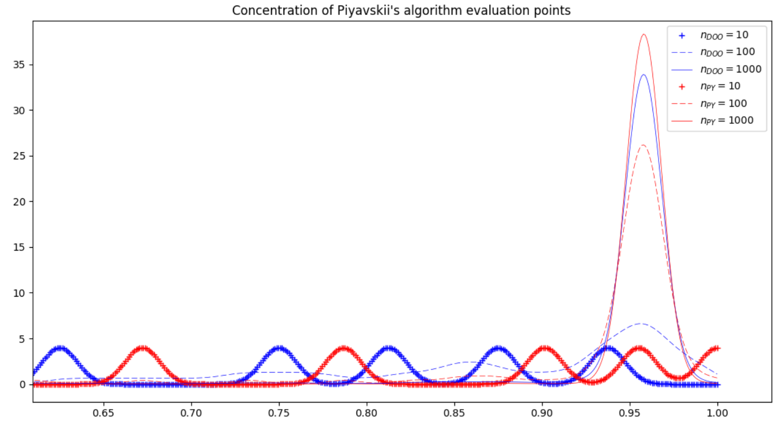


Figure 8: Kernel estimate of the density of PY and DOO evaluation points (Ackley test function with no Shift)

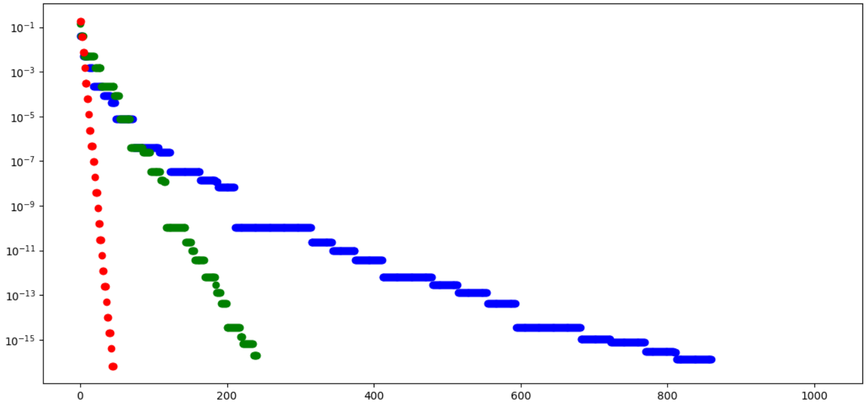


Figure 9: $d^{\star} = 0$ PY, DOO, DIRECT

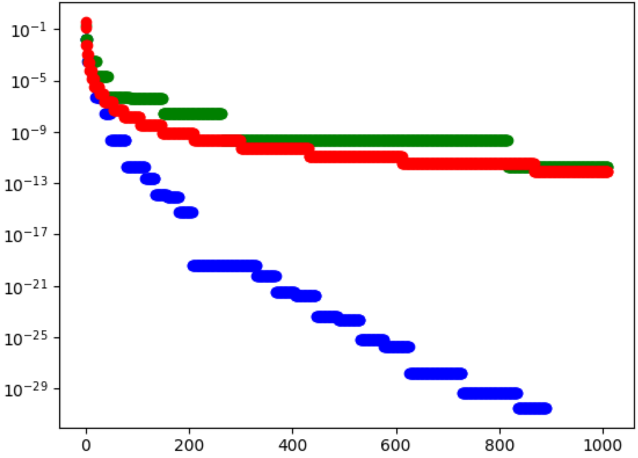


Figure 10: $d^{\star} = 1/2$ PY, DOO, DIRECT